Worksheet 3 Lists and linked lists

Task 1

1. 'Random Clothing Task' - Complete the following to show the operations implemented on a list of clothing items, initialised as an empty list `clothes[]`

Operation	List	Returns
isEmpty()		
len()		
append("socks")		
append("shoes")		
append("hat")		
append("socks")		
count("socks")		
index("shoes")		
len(clothes)		
insert(2, "gloves")		
remove("socks")		
pop()		
remove("shirt")		
append("socks")		
append("shorts")		
len(clothes)		
index("gloves")		
pop(1)		

Task 2

2. An unsorted list contains integers in the range 0-150. The following pseudocode has been written to count and print the number of integers that are in the range 80-100, and then to remove these numbers from the list and print the amended list.

```
list1 = [34,56,34,26,80,57,98,100,80,64,102,300,35,6,87,88]
count = 0
for index = 0 to (len(list1) – 1)
  if (list1[index] >=80) AND (list1[index] <=100) then
    count = count + 1
  endif
next index
print (“Number of integers in range 80-100”, count)

for index = 0 to (len(list1) – 1)
  if (list1[index] >=80) and (list1[index] <=100) then
        item = list1[index]
        list1.remove(item)
  endif
next index
print(list1)
```

When the program is coded and run, the first part works correctly but it crashes in the second FOR loop with the message

“if (list1[index] >=80) & (list1[index] <=100):

IndexError: list index out of range

Why does it crash?

Correct the pseudocode.

3. A program is to be written which merges the following two sorted lists **list1** and **list2** into a single sorted list called **mergeList** and prints out all three lists.

   list1 = [2,5,15,36,47,56,59,78,156,244,268]

   list2 = [18,39,42,43,66,69,100]

   (a) Which list functions will be useful in this program?

   (b) Write an algorithm to do this in ordinary English. You may find it useful to write the numbers from each list on pieces of paper and do the task manually, or use the bus cards from the previous lesson, split into two sorted lists of uneven length..

   (c) Convert the algorithm into pseudocode.

(d) Code and test the program in a programming language of your choice.

Task 3

4. A linked list abstract data type (ADT) has the following operations:

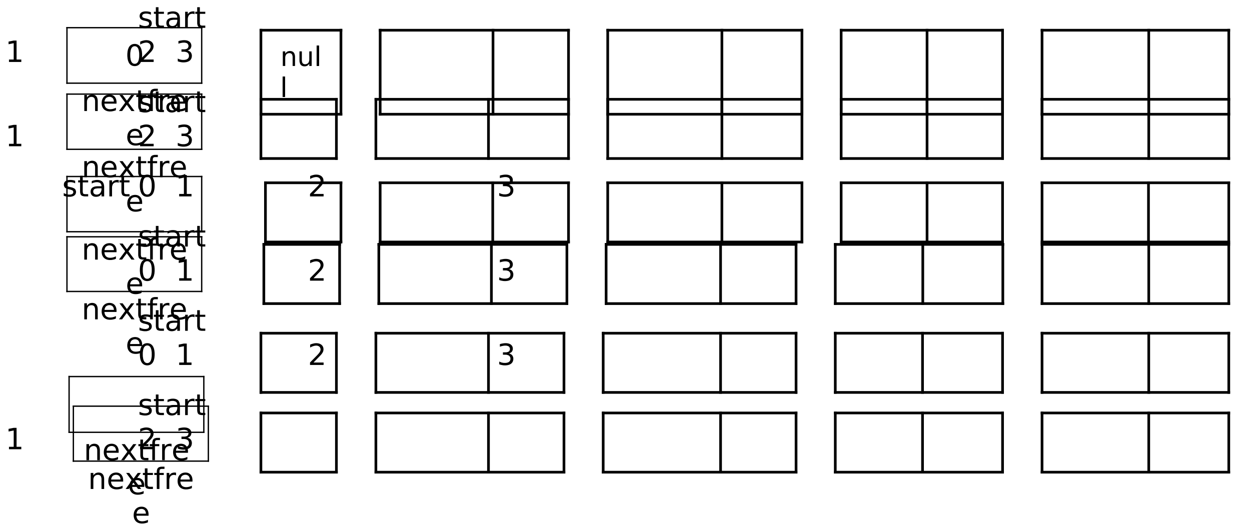
   - create linked list
   - add item to linked list
   - remove item from linked list

   Each node in the linked list consists of a name and a pointer to the next item in the linked list. Items are maintained in alphabetical order.

   A variable called `start` holds the index of the first item in the list

   (a) Show the state of the list after each of the following operations are carried out.

   ```
   CreateLinkedList
   AddItem(“Logan”)
   AddItem(“Poppy”)
   AddItem(“Ron”)
   DeleteItem(“Poppy”)
   AddItem(“James”
   ```



(b) The linked list is to be implemented as an array of 50 records called `myList`.

A node is defined as follows:

```
type nodeType
     string name
     integer pointer
endType


dim myList[0..49] of nodeType
```

The variable `pointer` holds the index of the next node. A variable called `nextfree` holds the index of the next free space in the array.The data in the linked list can be accessed in sequence by following the pointers to the next node.

The array is initialised using the following algorithm:

```
for index = 0 to 48
   myList[index].pointer = index + 1
next index
myList[49].pointer = null
start = null
nextfree = 0
```

Show the state of the linked list using the first diagram below, after initialisation of the array.

start =	**nextfree**

index	name	pointer
0		
1		
2		
3		
4		
:		
49		

start =	**nextfree**

index	name	pointer
0		
1		
2		
3		
4		
:		
49		

(c) Using the second diagram, show the state of the list after the following operations are carried out.

```
CreateLinkedList
AddItem(“Logan”)
AddItem(“Poppy”)
```

(d) Refer to the pseudocode on the next page.

(i) Fill in lines 3 and 4 to check for full list

(ii) What is the function of lines 7 - 11?

The procedure `AddItem(newItem)` is shown below.

```
01  procedure AddItem(newItem)
02  // check if list is full and if so, print error message
03
04
05    else
06      myList[nextfree].name = newName
07      if start = null then
08        temp = myList[nextfree].pointer       //save pointer
09        myList[nextfree].pointer = null
10        start = nextfree
11        nextfree = temp
12      else
13        p = start
14        if newName < myList[p].name then
15          myList[nextfree].pointer = start
16          start = nextfree
17        else
18          placeFound = false                 // general case
19          while myList[p].pointer <> null and placeFound = false
20            //peek ahead
21            if newName >= myList[myList[p].pointer].name then
22              p = myList[p].pointer
23            else
24              placefound = True
25            endif
26          endwhile
27          temp = nextFree
28          nextfree = node[nextfree].pointer
29          node[temp].pointer = node[p].pointer
30          node[p].pointer = temp
31        endif
32      endif
33    endif
```

```
34   endprocedure
```

(iii) What condition is line 14 of the pseudocode checking for?

(iv) Show the state of the list after three further operations:

```
AddItem(“Alan”)
DeleteItem(“Poppy”)
AddItem(“James”)
```

start =

nextfree

index	name	pointer
0		
1		
2		
3		
4		
:		
49		

5. Deleting an item from a linked list.
Here is an alphabetically ordered linked list, ListA, of animals. This implementation uses:

- a variable **start** to indicate the first item in the list
- a null in the pointer field to indicate the end of the list

index	animal	pointer
0	Snake	null
1	Dog	2
2	Mouse	0
3	Ant	1
4		5
5		Null

start = 3

nextfree
4

(a) (i) What is the value of `listA[start].pointer`?

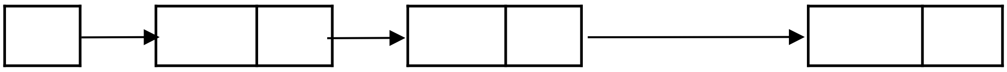
(ii) What is the value of `listA[listA[start].pointer].pointer`?

(iii) If `p = 1`, what is the value of `listA[listA[p].pointer].name`?

(b) The following pseudocode deletes an item in the table.

```
xName = “Mouse”
// check for empty list
if start = null then
print (“List is empty”)
else
  p = start
  if deleteName = listA[start].name then
    start = listA[start].pointer
  else
    while deleteName <> listA[listA[p].pointer].name
      p = listA[p].pointer
    endwhile
  endif
endif
nextptr = listA[p].pointer
listA[p].pointer = listA[nextptr].pointer
```

(i) Complete the diagram below to show the list after deleting Mouse according to the algorithm given in the pseudocode.



(ii) Complete the table below after deleting Mouse

(iii) What special case is line 7 of the pseudocode checking for?

index	animal	pointer
0		
1		
2		
3		
4		
5		

start = 3

nextfree
4

(iv) In the pseudocode given, the space left by the deleted item is not linked back into the list of free space. Explain how this could be done.

Show below what each node would hold if this was done.

index	animal	pointer
0		
1		
2		
3		
4		
5		

start = 3

nextfree